Probabilistic Programming Languages (PPL)

MPRI 2.40 - 2024/2025

Christine Tasson

Septembre, the 18th

Course

Thematics, teachers, notation

Probabilistic Programming

Thematics: Programming Languages

- Measure Theory & Statistical Inference Algorithms
- Semantics, Syntax & Implementation
- Static Analysis

Notation: 30% take-home assignment + 70% written exam

Material: MPRI 2.40 - Lectures material on github

Teachers

G. Baudart
IRIF - INRIA

X. Rival
DI - ENS

C. Tasson
LIP6 - Supaero

Introduction

Langage and Modeling

Probabilistic Programming

Paradigm that allows to model uncertainty through probability distribution and to handle them through stochastic process. Generation and inference are automated.

Specific Languages or libraries

Stan, Gen.jl, Pyro,...

Used for modeling in

- vision (image generation),
- robotics (planification),
- health (epidemiologie),
- social sciences (opinion poll),...

Today's schedule

Principles of probabilistic programming

Programs & simulation

Statistical learning, conditioning & bayesian inference

Modeling with $\mu-PPL$

https://github.com/gbdrt/mu-ppl

Examples & exercises

Mathematical tools

Random Variables

The Law of Large Numbers

Bayes formula and importance sampling

Principles of PPL

Enumeration

What does the program model ?

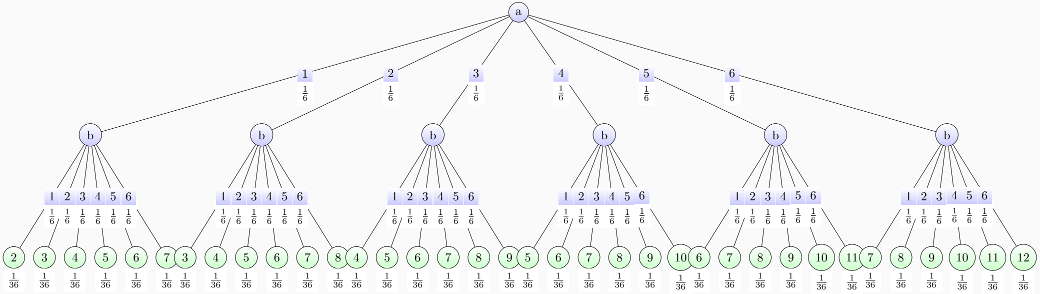
```
def dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    return a + b
```

What does the program model ?

```
def dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    return a + b
```

It simulates the **random variable** dice:
"The sum of the values output by the two independent fair dice."

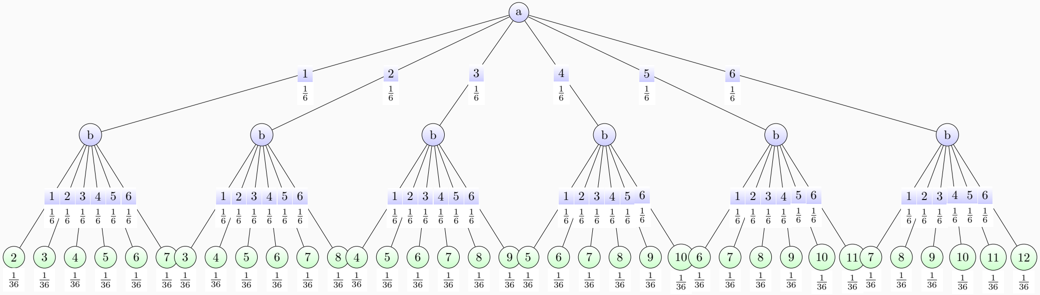
At each execution, the operator sample **simulates** a random variable with Uniform law on $\{1, \dots, 6\}$.



How to compute the law of the random variable associated to this program ?

```
def dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    return a + b

with Enumeration():
    dist: Categorical[int] = infer(dice)
```

The law dist of the random variable dice can be computed by **enumeration** of all the possible samples.

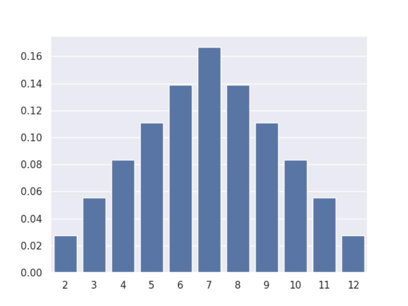


What is the law of the random variable associated to this program ?

```
def dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    return a + b

with Enumeration():
    dist: Categorical[int] = infer(dice)
    print(dist.stats())
    viz(dist)
    plt.show()
```

The discrete law:

$$
\begin{aligned}
[\![\text{dice}]\!] : &= \mathbb{P}(\text{dice}() = k) \\
&= \sum_{a=1}^{6} \sum_{b=1}^{6} \tfrac{1}{36} \mathbb{1}_{\{a+b=k\}}
\end{aligned}
$$



What does the program model ?

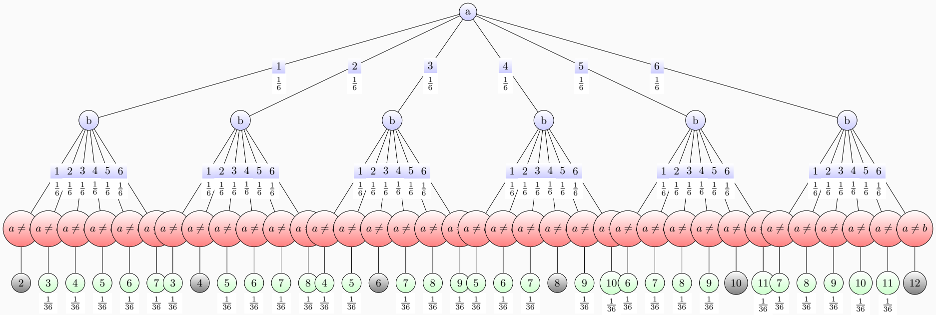
```
def hard_dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    assume (a != b)
    return a + b
```

What does the program model ?

```
def hard_dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    assume (a != b)
    return a + b
```

It simulates the **random variable**:
"The sum of two independent fair dice, given that the output value are distinct."

The operator **assume** rejects samples that do not satisfy the wanted property.

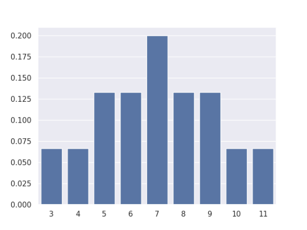


What is the law associated to the program ?

```
def hard_dice() → int:
    a = sample(RandInt(1, 6), name="a")
    b = sample(RandInt(1, 6), name="b")
    assume (a != b)
    return a + b

with Enumeration():
    dist: Categorical[int] = infer(hard_dice)
    print(dist.stats())
    viz(dist)
    plt.show()
```

The conditional law

$$[\![\text{hard\_dice}]\!](k) = \mathbb{P}(a + b = k \mid a \neq b) = \frac{\sum_{a \neq b \in \{1,\dots,6\}^2} \frac{1}{36} \mathbb{1}_{\{a+b=k\}}}{\sum_{a \neq b \in \{1,\dots,6\}^2} \frac{1}{36}}$$



Principles of PPL

Monte-Carlo Simulation

What does the program model ?

```
def disk() → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    assume (x**2 + y**2 < 1)
    return (x, y)
```

What does the program model ?

```
def disk() → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    assume (x**2 + y**2 < 1)
    return (x, y)
```

It simulates the **random variable** of conditional law: *"The position sampled in a uniform over a square, given that its distance to the center is strictly less than* 1*."*

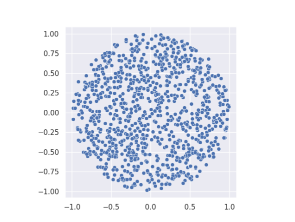
How to compute the law associated to this program ?

```
def disk() → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    assume (x**2 + y**2 < 1)
    return (x, y)

with RejectionSampling(num_samples=1000):
    dist: Empirical = infer(disk)
    x, y = zip(*dist.samples)
    sns.scatterplot(x=x, y=y)
```

Monte-Carlo simulation:

N-sample: $(X_1, \ldots, X_N)$ *i.i.d.*

with same law as disk



The law of large numbers The mean is the limit of the empirical mean of the N-sample:

$$\frac{1}{N}\sum_{i=1}^{N} g(X_i) \xrightarrow[N\to\infty]{} \mathbb{E}(g(X)).$$

The characteristics of the law of disk can be approximated.

What does the program model ?

```
def position(o: float) → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    d2 = x**2 + y**2
    observe(Gaussian(d2, 0.1), o)
    return (x, y)
```

What does the program model ?

```
def position(o: float) → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    d2 = x**2 + y**2
    observe(Gaussian(d2, 0.1), o)
    return (x, y)
```

The program position models the law of "the position of a point in a square given the *noisy* measure of the square of the distance to the center."

Given the position (X, Y) **a priori** uniform on $[-1, 1]^2$,
let us **observe** the random variable Z gaussian centered on the square of the distance to the center to (X, Y) and get the **likelyhood** of this position.

The program position models the law of (X, Y) given $Z = o$.

How to compute the law associated to this program ?

```
def position(o: float) → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    d2 = x**2 + y**2
    observe(Gaussian(d2, 0.1), o)
    return (x, y)

with ImportanceSampling(num_particles=10000):
    dist: Categorical = infer(position, 0.5)
```

Prior

$$(X, Y) \sim \text{Uniform}([-1, 1]^2)$$

Likelihood

$$Z \sim \text{Gaussian}(X^2 + Y^2, 0.1)$$

Posterior: position

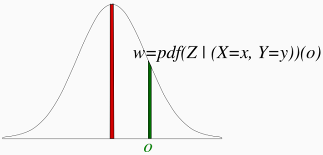
(X, Y) given $Z = o$.

Monte-Carlo simulation

N-sample *i.i.d.* with prior law (X, Y)

observe scores each execution with its likelihood: the density of Z at o given the position $(X, Y) = (x, y)$

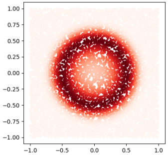
the law of large numbers ensures that we can approximates the characteristics of the posterior (X, Y) given $Z = o$



How to compute the law associated to this program ?

```
def position(o: float) → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    d2 = x**2 + y**2
    observe(Gaussian(d2, 0.1), o)
    return (x, y)

with ImportanceSampling(num_particles=10000):
    dist: Categorical = infer(position, 0.5)
```

Monte-Carlo simulation:
N-sample *i.i.d.*: $(X_i, Y_i)_{i \leq N}$
scored with
$w_i = \mathbb{P}(Z = o | (X_i, Y_i) = (x_i, y_i))$



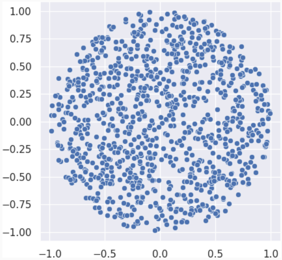
Law of large numbers: the mean is the limit of the empirical mean of N-samples:

$$\frac{\frac{1}{N}\sum_i g(X_i, Y_i) w_i}{\frac{1}{N}\sum_i w_i} \xrightarrow[N\to\infty]{} \frac{\mathbb{P}(Z = o | (X_i, Y_i) = (x, y))}{\mathbb{P}(Z = o)} = \mathbb{E}(g(X, Y) | Z = o).$$

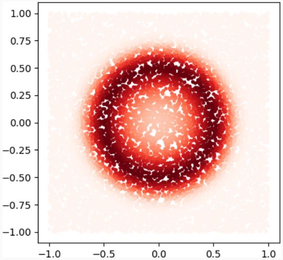
Conditioning: hard by reject and soft by scoring)

```
def disk() → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    d2 = x**2 + y**2
    assume (d2 < 1)
    return (x, y)

with RejectionSampling(num_samples=10000):
    dist: Empirical = infer(disk)
```

```
def position(o: float) → Tuple[float, float]:
    x = sample(Uniform(-1, 1))
    y = sample(Uniform(-1, 1))
    d2 = x**2 + y**2
    observe(Gaussian(d2, 0.1), o)
    return (x, y)

with ImportanceSampling(num_particles=10000):
    dist: Categorical = infer(position, 0.5)
```



Monte-Carlo Simulation



Principles of PPL

Hierarchical Models

What does the program model ?

```
def success(s:int) → int:
    n = sample(RandInt(10, 20))
    d_success = Binomial(n, 0.5)
    observe(d_success, s)
    return n
```

What does the program model ?

```
def success(s:int) → int:
    n = sample(RandInt(10, 20))
    d_success = Binomial(n, 0.5)
    observe(d_success, s)
    return n
```

Prior law X is uniform from 10 to 20

Conditional law S the number of success among $X = x$ flips of a fair coin

Likelyhood

$\mathbb{P}(S = s | X = n) = \mathrm{Binomial(n,\ 0.5)}(s)$

The program success models the **posterior** law: the conditional law of X given $S = s$.

"How many times (between 10 and 20) the fair coin has been tossed given there was s success?"

Exercises

Modeling in $\mu{-}PPL$

Exercises

μ−*PPL* is a micro probabilistic programming language:

https://github.com/gbdrt/mu-ppl

that you can **install** with the command:

$ pip install git+https://github.com/gbdrt/mu-ppl

The exercises can be done

- on paper (as during the exam),
- on your machine locally via a notebook
- or on Google Collab.

Mathematical tools

Random Variables

Measurable Space and Random Variables

Measurable space $(\Omega, \mathcal{F})$

A universe Ω is a set of possible results. A σ-algebra is a set $\mathcal{F}$ of parts of Ω such that

$$\emptyset \in \mathcal{F} \text{ and if } A_1, A_2, \ldots \in \mathscr{F}, \text{ then } \bigcup_{i=1}^{\infty} A_i \in \mathscr{F} \text{ and if } A \in \mathscr{F}, \text{ then } A^c \in \mathscr{F}.$$

A **measure** on $(\Omega, \mathscr{F})$ is a function $\mu : \mathscr{F} \to [0, 1]$ such that

$$\mu(\emptyset) = 0 \;\text{ and if } A_1, A_2, \ldots \in \mathscr{F} \text{ are pairwise disjoint, then } \mu\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mu(A_i).$$

A probability measure $\mathbb{P}$ is a measure such that $\mathbb{P}(\Omega) = 1$.

Random variable $X : \Omega \to \mathbb{R}^n$ such that

$$\forall x \in \mathbb{R}^n \; \{\omega \in \Omega \mid X(\omega) \leq x\} \in \mathscr{F}.$$

The probability distribution μ of X, denoted $X \sim \mu$ is the pushforward probability measure of X on its outputs:

$$\forall U \subseteq \mathbb{R}^n, \; \mu(U) = \mathbb{P}(X^{-1}(U)) = \mathbb{P}(\{\omega \in \Omega \mid X(\omega) \in U\})$$

Discrete Random Variable

Definition
A random variable $X : \Omega \to \mathbb{R}$ is discrete if it take a countable number of values.

The probability mass function (PMF) of a discrete random variable X is a function $f : \mathbb{R} \to [0, 1]$ such that: $f(x) = \mathbb{P}(X = x) = \mathbb{P}(\{\omega \in \Omega \mid X(\omega) = x\})$.

Discrete Random Variable

Definition
A random variable $X : \Omega \to \mathbb{R}$ is discrete if it take a countable number of values.

The probability mass function (PMF) of a discrete random variable X is a function $f : \mathbb{R} \to [0, 1]$ such that: $f(x) = \mathbb{P}(X = x) = \mathbb{P}(\{\omega \in \Omega \mid X(\omega) = x\})$.

The mean (expected value), written $\mathbb{E}(X)$ of a discrete random variable X is:

$$\mathbb{E}(X) = \sum_x x\,\mathbb{P}(X = x) = \sum_x x\,f(x)$$

when the sum exists (it is always the case when $X > 0$).

Discrete Random Variable

Definition
A random variable $X : \Omega \to \mathbb{R}$ is discrete if it take a countable number of values.

The probability mass function (PMF) of a discrete random variable X is a function $f : \mathbb{R} \to [0, 1]$ such that: $f(x) = \mathbb{P}(X = x) = \mathbb{P}(\{\omega \in \Omega \mid X(\omega) = x\})$.

The mean (expected value), written $\mathbb{E}(X)$ of a discrete random variable X is:

$$\mathbb{E}(X) = \sum_x x \, \mathbb{P}(X = x) = \sum_x x \, f(x)$$

when the sum exists (it is always the case when $X > 0$).

If $g : \mathbb{R} \to \mathbb{R}$ measurable and X is a discrete random variable, then $g(X)$ is a discrete random variable

$$\mathbb{E}(g(X)) = \sum_x g(x) \, \mathbb{P}(X = x)$$

How to represent Discrete Probability Distributions

Data Structure:

- **Support** : output values
- **Mass functions** : probability associated to each value
- **Mean** : when there is an analytic value

Operations:

- **Random generator** i.e. sampling
- **Approximated expected value** given or computed by Monte-Carlo simulation

Example of discrete random variables

Bernoulli $X \sim \mathcal{B}(p)$
Toss of a coin of bias p.

Binomial $X \sim \mathrm{Bin}(n, p)$
Sum of toss of n coins of bias p.

Empirical/Uniform $X \sim \mathrm{Uniform}(xs)$
Every value in xs is equally likely.

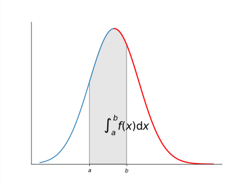
Categorical $X \sim \mathcal{C}(xs, ps)$
Value $xs[i]$ has probability $ps[i]$.

Continuous random variable with density

Continuous random variable with density

Definition
A random variable $X : \Omega \to \mathbb{R}$ is continuous with density (absolutely continuous with respect to Lebesgues measure) if its cumulative distribution function: $F(x) = \mathbb{P}(\{\omega \mid X(\omega) \leq x\})$ can be described by the integral of $f : \mathbb{R} \to [0, \infty]$ called the probability density function (PDF) of X:

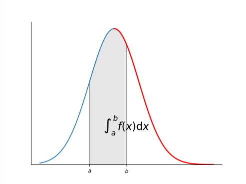
$$F(x) = \mathbb{P}(X \leq x\}) = \int_{-\infty}^{x} f(u)du$$



Continuous random variable with density

Definition
A random variable $X : \Omega \to \mathbb{R}$ is continuous with density (absolutely continuous with respect to Lebesgues measure) if its cumulative distribution function: $F(x) = \mathbb{P}(\{\omega \mid X(\omega) \leq x\})$ can be described by the integral of $f : \mathbb{R} \to [0, \infty]$ called the probability density function (PDF) of X:

$$F(x) = \mathbb{P}(X \leq x\}) = \int_{-\infty}^{x} f(u)du$$



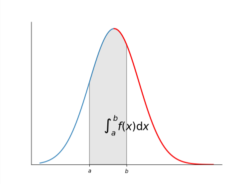
The mean, denoted $\mathbb{E}(X)$ of the continuous random variable X is defined by the integral when it exists:

$$\mathbb{E}(X) = \int_{-\infty}^{\infty} x\ f(u)du$$

Continuous random variable with density

Definition
A random variable $X : \Omega \to \mathbb{R}$ is continuous with density (absolutely continuous with respect to Lebesgues measure) if its cumulative distribution function: $F(x) = \mathbb{P}(\{\omega \mid X(\omega) \leq x\})$ can be described by the integral of $f : \mathbb{R} \to [0, \infty]$ called the probability density function (PDF) of X:

$$F(x) = \mathbb{P}(X \leq x\}) = \int_{-\infty}^{x} f(u)du$$



The mean, denoted $\mathbb{E}(X)$ of the continuous random variable X is defined by the integral when it exists:

$$\mathbb{E}(X) = \int_{-\infty}^{\infty} x\ f(u)du \qquad \mathbb{E}(g(X)) = \int_{-\infty}^{\infty} g(u)\ f(u)du$$

How to represent ~~Discrete~~ Continuous Probability Distributions

Data Structure:

- **Support** : potential output values
- ~~Mass~~ **Probability density function** : probability associated to each interval
- **Mean** : when there is an analytic value

Operations:

- **Random generator** i.e. sampling
- **Approximated expected value** given or computed by Monte-Carlo simulation

Example of continuous random variables

Uniform $X \sim \mathcal{U}(a, b)$
with density function $f_X(x) = \frac{1}{b-a}$ if $a < x < b$, 0 otherwise.

Gaussian $X \sim \mathcal{N}(m, \sigma)$
with density function $f_X \frac{1}{\sigma\sqrt{2\pi}} \exp -\frac{1}{2}(\frac{x-\mu}{\sigma})^2$.

Mathematical tools

The Law of Large Number

Sample mean and law of large numbers

Sample mean: The sample mean of a random variable is obtained by simulation:

compute n samples denoted $x_1, \dots, x_n$.

compute the mean $\frac{1}{n}(x_1 + \cdots + x_n)$.

Random generator: it is sufficient to have a random generator for categorical substitutions with support $\{1, \dots, n\}$.

Sample mean and law of large numbers

Sample mean: The sample mean of a random variable is obtained by simulation:

compute n samples denoted $x_1, \ldots, x_n$.

compute the mean $\frac{1}{n}(x_1 + \cdots + x_n)$.

Random generator: it is sufficient to have a random generator for categorical substitutions with support $\{1, \ldots, n\}$.

Law of large numbers: The sample mean approximate the mean.

If $X_1, \ldots, X_n$ are independent identically distributed (i.i.d.) and the mean of $g(X)$ is finite, then

$$\frac{1}{n} \sum_{i=1}^{n} g(X_i) \to \mathbb{E}(g(X))$$

Sample mean and law of large numbers

Sample mean: The sample mean of a random variable is obtained by simulation:

compute n samples denoted $x_1, \dots, x_n$.

compute the mean $\frac{1}{n}(x_1 + \dots + x_n)$.

Random generator: it is sufficient to have a random generator for categorical substitutions with support $\{1, \dots, n\}$.

Law of large numbers: The sample mean approximate the mean.

If $X_1, \dots, X_n$ are independent identically distributed (i.i.d.) and the mean of $g(X)$ is finite, then

$$\frac{1}{n}\sum_{i=1}^{n} g(X_i) \to \mathbb{E}(g(X)) \qquad \frac{1}{n}\sum_{i=1}^{n} X_i \to \mathbb{E}(X)$$

Monte-Carlo simulation and the law of large numbers

A **Probabilistic program** is the random variable that values are the outcome of the program execution. If the program has no **effect**, then its executions are i.i.d. Then:

Monte-Carlo simulation and the law of large numbers

A **Probabilistic program** is the random variable that values are the outcome of the program execution. If the program has no **effect**, then its executions are i.i.d. Then:

Law of large numbers:

- Run n times the probabilistic program
- Store the outputs $x_1, \dots, x_n$.
- Compute $\frac{x_1 + \dots + x_n}{n}$ that approximates $\mathbb{E}(X)$ and $\frac{g(x_1) + \dots + g(x_n)}{n}$ that approximates $\mathbb{E}(g(x))$

Monte-Carlo simulation and the law of large numbers

A **Probabilistic program** is the random variable that values are the outcome of the program execution. If the program has no **effect**, then its executions are i.i.d. Then:

Law of large numbers:

Run n times the probabilistic program

Store the outputs $x_1, \dots, x_n$.

Compute $\frac{x_1+\cdots+x_n}{n}$ that approximates $\mathbb{E}(X)$ and $\frac{g(x_1)+\cdots+g(x_n)}{n}$ that approximates $\mathbb{E}(g(x))$

Monte-Carlo Simulation: histograms approximate distributions:

For a random variable X,

$\frac{1}{n}\#(\{i|x_i = x\})$ approximates $\mathbb{E}(\chi_{X=x}) = \mathbb{P}(X = x)$

$\frac{1}{n}\#(\{i|a \leq x_i \leq b\})$ approximates $\mathbb{E}(\chi_{a\leq X\leq b}) = \mathbb{P}(a \leq X \leq b)$

Mathematical tools

Bayes Law

Marijuana and the army

During Vietnam war in 1975, how to evaluate the proportion of smokers.

Experiment: To the question ”do you smoke ?”, a soldier toss a coin:

> If the coin is HEADS, she answers yes (even if she does not smoke),
> If the coin is TAIL, she answers the truth.

Thus, the answer can be yes either if the output is HEADS or if she smokes.

Observed Results: 200 answers, 160 yes and 40 no

What is the probability that a soldier smoke ?

Marijuana and the army

During Vietnam war in 1975, how to evaluate the proportion of smokers.

Experiment: To the question "do you smoke ?", a soldier toss a coin:

> If the coin is HEADS, she answers yes (even if she does not smoke),
> If the coin is TAIL, she answers the truth.

Thus, the answer can be yes either if the output is HEADS or if she smokes.

Observed Results: 200 answers, 160 yes and 40 no

What is the probability that a soldier smoke ? 60%

Marijuana and the army - privacy

During Vietnam war in 1975, how to evaluate the proportion of smokers.

Experiment: To the question ”do you smoke ?”, a soldier toss a coin:

If the coin is HEADS, she answers yes (even if she does not smoke),
If the coin is TAIL, she answers the truth.

Thus, the answer can be yes either if the output is HEADS or if she smokes.

Observed Results: 200 answers, 160 yes and 40 no

What is the probability that a soldier smoke ? 60%

What is the probability that a soldier smokes if she answered yes ?

Marijuana and the army - privacy

During Vietnam war in 1975, how to evaluate the proportion of smokers.

Experiment: To the question "do you smoke ?", a soldier toss a coin:

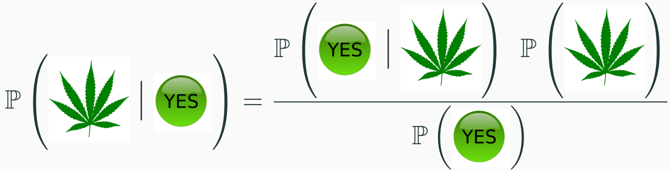
If the coin is HEADS, she answers yes (even if she does not smoke),
If the coin is TAIL, she answers the truth.

Thus, the answer can be yes either if the output is HEADS or if she smokes.

Observed Results: 200 answers, 160 yes and 40 no

What is the probability that a soldier smoke ? 60%

What is the probability that a soldier smokes if she answered yes ?



Conditional Probability

Definition
The conditional probability is defined when $\mathbb{P}(B) > 0$, then

$$\mathbb{P}(A \mid B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}.$$

Conditional Probability

Definition
The conditional probability is defined when $\mathbb{P}(B) > 0$, then

$$\mathbb{P}(A \mid B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}.$$

Two events A and B are independent when one hence all equivalent properties are satisfied $\mathbb{P}(A) = \mathbb{P}(A|B)$ or $\mathbb{P}(B) = \mathbb{P}(B|A)$ or $\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$.

Conditional Probability

Definition
The conditional probability is defined when $\mathbb{P}(B) > 0$, then

$$\mathbb{P}(A \mid B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}.$$

Two events A and B are independent when one hence all equivalent properties are satisfied $\mathbb{P}(A) = \mathbb{P}(A|B)$ or $\mathbb{P}(B) = \mathbb{P}(B|A)$ or $\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$.

Properties

if $\mathbb{P}(B) > 0$, then $\mathbb{P}(A) = \mathbb{P}(A|B)P(B) + \mathbb{P}(A|B^c)P(B^c)$.

if $B_1, B_2, \ldots, B_n$ is a partition Ω, then

$$\mathbb{P}(A) = \sum_{i=1}^{n} \mathbb{P}(A|B_i)\mathbb{P}(B_i)$$

Interlude theoretic - Bayes Law

Bayes Law

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(A|B)\,\mathbb{P}(B)}{\mathbb{P}(A)}$$

Bayes Formula

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(A|B)\,\mathbb{P}(B)}{\mathbb{P}(A|B)\,\mathbb{P}(B) + \mathbb{P}(A|B^c)\,\mathbb{P}(B^c)}$$

Generalization, if $\mathbb{P}(A) > 0$ and $B_1, B_2, \ldots, B_n$ is a partition of Ω such that $\forall i\, \mathbb{P}(B_i) > 0$, then

$$\mathbb{P}(B_j|A) = \frac{\mathbb{P}(A|B_j)P(B_j)}{\sum_{i=1}^{n} \mathbb{P}(A|B_i)\mathbb{P}(B_i)}$$

Bayes Law - Programs tests and bugs

Probabilistic Programming and Bayesian Methods for Hackers, C. Davidson-Pilon.

A programmer writes code and tests and wonders: Are there bugs in my code?

- A is the event there is NO bug and $\mathbb{P}(A) = p$.
- X is the event all tests are passed.
- assume that there is a 50% chance that the tests are passed given that there is a bug.

What is the probability $\mathbb{P}(A|X)$ of the event there is no bug given that all the tests are passed ?

Bayes Law - Programs tests and bugs

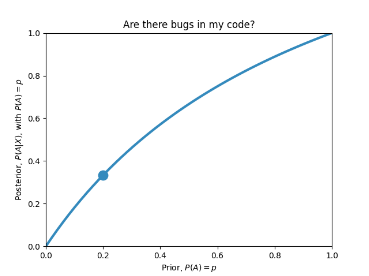
Probabilistic Programming and Bayesian Methods for Hackers, C. Davidson-Pilon.

A programmer writes code and tests and wonders: Are there bugs in my code?

- A is the event there is NO bug and $\mathbb{P}(A) = p$.
- X is the event all tests are passed.
- assume that there is a 50% chance that the tests are passed given that there is a bug.

What is the probability $\mathbb{P}(A|X)$ of the event there is no bug given that all the tests are passed ?

$$
\begin{aligned}
\mathbb{P}(X) &= \mathbb{P}(X, A) + \mathbb{P}(X, \overline{A}) \\
&= \mathbb{P}(X|A)\mathbb{P}(A) + \mathbb{P}(X|\overline{A})\mathbb{P}(\overline{A}) \\
&= 1 \cdot p + 0.5 \cdot (1 - p) = \frac{p+1}{2} \\
\mathbb{P}(A|X) &= \frac{\mathbb{P}(X|A)\mathbb{P}(A)}{\mathbb{P}(X)} = \frac{2p}{1+p}
\end{aligned}
$$



Conditional Probability and random variables

Discrete

Bayes Law

$$\mathbb{P}(Y = y | X = x) = \frac{\mathbb{P}(X = x | Y = y)\,\mathbb{P}(Y = y)}{\mathbb{P}(X = x)}$$

Conditional Probability and random variables

Discrete

Bayes Law

$$\mathbb{P}(Y = y | X = x) = \frac{\mathbb{P}(X = x | Y = y)\,\mathbb{P}(Y = y)}{\mathbb{P}(X = x)}$$

Bayes Formula

$$\mathbb{P}(Y = y | X = x) = \frac{\mathbb{P}(X = x | Y = y)\,\mathbb{P}(Y = y)}{\sum_b \mathbb{P}(X = x | Y = b)\,\mathbb{P}(Y = b)}$$

Conditional Probability and random variables

Discrete

Bayes Law

$$\mathbb{P}(Y = y|X = x) = \frac{\mathbb{P}(X = x|Y = y)\,\mathbb{P}(Y = y)}{\mathbb{P}(X = x)}$$

Bayes Formula

$$\mathbb{P}(Y = y|X = x) = \frac{\mathbb{P}(X = x|Y = y)\,\mathbb{P}(Y = y)}{\sum_b \mathbb{P}(X = x|Y = b)\,\mathbb{P}(Y = b)}$$

Continuous

Bayes Law

$$f_{Y|X}(y|x) = \frac{f_{X|Y}(x|y)\,f_Y(y)}{f_X(x)}$$

Conditional Probability and random variables

Discrete

Bayes Law

$$\mathbb{P}(Y=y|X=x)=\frac{\mathbb{P}(X=x|Y=y)\,\mathbb{P}(Y=y)}{\mathbb{P}(X=x)}$$

Bayes Formula

$$\mathbb{P}(Y=y|X=x)=\frac{\mathbb{P}(X=x|Y=y)\,\mathbb{P}(Y=y)}{\sum_b \mathbb{P}(X=x|Y=b)\,\mathbb{P}(Y=b)}$$

Continuous

Bayes Law

$$f_{Y|X}(y|x)=\frac{f_{X|Y}(x|y)\,f_Y(y)}{f_X(x)}$$

Bayes Formula

$$f_{Y|X}(y|x)=\frac{f_{X|Y}(x|y)\,f_Y(y)}{\int_{-\infty}^{\infty} f_{X|Y}(x|y)\,f_Y(y)dy}$$

Theoretical Interlude - Conditional Law

Joint Distribution: Let X and Y be two random variables. Their joint distribution function is given by

$$F(x, y) = \mathbb{P}(X \leq x, Y \leq y)$$

X and Y are jointly continuous with density if there is joint probability density function f such that

$$F(x, y) = \int_{-\infty}^{y} \int_{-\infty}^{x} f_{X,Y}(u, v) du dv$$

Theoretical Interlude - Conditional Law

Joint Distribution: Let X and Y be two random variables. Their joint distribution function is given by

$$F(x, y) = \mathbb{P}(X \leq x, Y \leq y)$$

X and Y are jointly continuous with density if there is joint probability density function f such that

$$F(x, y) = \int_{-\infty}^{y} \int_{-\infty}^{x} f_{X,Y}(u, v) du dv$$

Then, the second marginal give the density function of the law of Y

$$f_Y(y) = \int_{-\infty}^{x} f_{X,Y}(u, y) du$$

Theoretical Interlude - Conditional Law

Joint Distribution: Let X and Y be two random variables. Their joint distribution function is given by

$$F(x,y) = \mathbb{P}(X \leq x, Y \leq y)$$

X and Y are jointly continuous with density if there is joint probability density function f such that

$$F(x,y) = \int_{-\infty}^{y} \int_{-\infty}^{x} f_{X,Y}(u,v)dudv$$

Then, the second marginal give the density function of the law of Y

$$f_Y(y) = \int_{-\infty}^{x} f_{X,Y}(u,y)du$$

Let us define **conditional density function** by:

$$f_{X|Y}(x|y) = \frac{f_{X,Y}(x,y)}{f_Y(y)}$$

Mathematical tools

Importance Sampling

How to compute the law associated to this program ?

```
def success(s:int) → int:
    n = sample(RandInt(10, 20))
    observe(Binomial(n, 0.5), s)
    return n

with ImportanceSampling(num_particles=100):
    dist: Categorical = infer(success, 10)
```

Monte-Carlo simulation:
N-sample $(X_1, \ldots, X_N)$ independent identically distributed with law X uniform on 10 to 20.
Weight: likelyhood
$w_i = \mathbb{P}(S = s | X_i = n_i)$

Bayes Formula:

$$\mathbb{P}(X = n | S = s) = \frac{\mathbb{P}(S = s | X = n)\ \mathbb{P}(X = n)}{\mathbb{P}(S = s)} = \frac{\mathbb{P}(S = s | X = n)\ \mathbb{P}(X = n)}{\displaystyle\sum_{k=10}^{20} \mathbb{P}(S = s | X = k)\mathbb{P}(X = k)}$$

Law of Large Number: The mean is the limit of the empirical mean:

$$\frac{\sum_{i=1}^{N} g(X_i) w_i}{\sum_{i=1}^{N} w_i} \xrightarrow[N \to \infty]{} \mathbb{E}(g(X) | S = s).$$

Mean approximation

Loi des grands nombres: $h(X_i) = w_i = \mathbb{P}(S = s|X_i)$

$$\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} h(X_i) = \mathbb{E}(h(X)) = \sum_{k=10}^{20} \mathbb{P}(S = s|X = k)\mathbb{P}(X = k) = \mathbb{P}(S = s)$$

Loi des grands nombres: $\rho(X_i) = \frac{w_i g(X_i)}{\mathbb{P}(S=s)}$

$$\begin{aligned} \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \rho(X_i) &= \mathbb{E}(\rho(X)) = \sum_{k=10}^{20} \frac{\mathbb{P}(S = s|X = k)\mathbb{P}(X = k)g(k)}{\mathbb{P}(S = s)} \\ &= \sum_{k=10}^{20} \mathbb{P}(X = k|S = s)g(k) = \mathbb{E}(g(X)|S = s) \end{aligned}$$

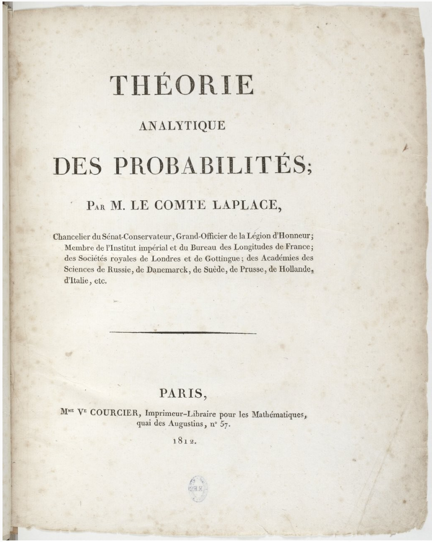
Conclusion:

$$\frac{\sum g(X_i)w_i}{\sum w_i} \xrightarrow[N\to\infty]{} \mathbb{E}(g(X)|S = s).$$

Conclusion

A historical example

Laplace problem



THÉORIE
ANALYTIQUE
DES PROBABILITÉS;

PAR M. LE COMTE LAPLACE,

Chancelier du Sénat-Conservateur, Grand-Officier de la Légion d'Honneur; Membre de l'Institut impérial et du Bureau des Longitudes de France; des Sociétés royales de Londres et de Gottingue; des Académies des Sciences de Russie, de Danemarck, de Suède, de Prusse, de Hollande, d'Italie, etc.

PARIS,
Mme Ve COURCIER, Imprimeur-Libraire pour les Mathématiques, quai des Augustins, n° 57.
1812.

Source gallica.bnf.fr / Bibliothèque nationale de France

What is the probability that the proportion of boys over girls is greater in Paris than in London in the XVIIIth century?

DES PROBABILITÉS. 377

28. C'est principalement aux naissances, que l'analyse précédente est applicable, et l'on peut en déduire non-seulement pour l'espèce humaine, mais pour toutes les espèces d'êtres organisés, des résultats intéressans. Jusqu'ici les observations de ce genre n'ont été faites en grand nombre, que sur l'espèce humaine : nous allons soumettre au calcul, les principales.

Considérons d'abord les naissances observées à Paris, à Londres, et dans le royaume de Naples. Dans l'espace des 40 années écoulées depuis le commencement de 1475, époque où l'on a commencé à distinguer à Paris, sur les registres, les naissances des deux sexes, jusqu'à la fin de 1784, on a baptisé dans cette capitale, 393386 garçons, et 377555 filles, les enfans trouvés étant compris dans ce nombre : cela donne à peu près $\frac{25}{24}$ pour le rapport des baptêmes des garçons à ceux des filles.

Dans l'espace des 95 années écoulées depuis le commencement de 1664 jusqu'à la fin de 1758, il est né à Londres, 737629 garçons, et 698958 filles ; ce qui donne $\frac{19}{18}$ à peu près, pour le rapport des naissances des garçons à celles des filles.

Enfin, dans l'espace des neuf années écoulées depuis le commencement de 1774 jusqu'à la fin de 1782, il est né dans le royaume de Naples, la Sicile non-comprise, 782352 garçons, et 746821 filles ; ce qui donne $\frac{22}{21}$ pour le rapport des naissances des garçons à celles des filles.

Les plus petits de ces nombres de naissances, sont relatifs à Paris; d'ailleurs, c'est dans cette ville que les naissances des garçons et des filles, approchent le plus de l'égalité. Par ces deux raisons, la probabilité que la possibilité de la naissance d'un garçon surpasse $\frac{1}{2}$, doit y être moindre qu'à Londres et dans le royaume de Naples. Déterminons numériquement cette probabilité.

Nommons p le nombre des naissances masculines observées à Paris, q celui des naissances féminines, et x la possibilité d'une naissance masculine, c'est-à-dire la probabilité qu'un enfant qui doit naître, sera un garçon; $1-x$ sera la possibilité d'une naissance féminine, et l'on aura la probabilité que sur $p+q$ naissances,

48

. . .

384 THÉORIE ANALYTIQUE

la probabilité que la possibilité des baptêmes des garçons est plus grande à Londres qu'à Paris, a pour expression,

$$1-\frac{i.c^{-\frac{1}{2i^2}}}{\sqrt{2\pi}}\cdot\cfrac{1}{1+\cfrac{i^2}{1+\cfrac{2i^2}{1+\cfrac{3i^2}{1+\cfrac{4i^2}{1+\text{etc.}}}}}}\;.\qquad(\mu)$$

En faisant dans cette formule,

$$p=393386\,,\quad q=377555\,,$$
$$p'=737629\,,\quad q'=698958\,,$$

elle devient

$$1-\frac{1}{328269}.$$

Il y a donc 328268 à parier contre un, qu'à Londres, la possibilité des baptêmes des garçons est plus grande qu'à Paris. Cette probabilité approche tellement de la certitude, qu'il y a lieu de rechercher la cause de cette supériorité.

Parmi les causes qui peuvent la produire, il m'a paru que les baptêmes des enfans trouvés, qui font partie de la liste annuelle des baptêmes à Paris, devaient avoir une influence sensible sur le rapport des baptêmes des garçons à ceux des filles ; et qu'ils devaient diminuer ce rapport, si, comme il est naturel de le croire, les parens des campagnes environnantes, trouvant de l'avantage à retenir près d'eux les enfans mâles, en avaient envoyé à l'hospice des Enfans trouvés de Paris, dans un rapport moindre que celui des naissances des deux sexes. C'est ce que le relevé des registres de cet hospice m'a fait voir avec une très-grande probabilité. Depuis le commencement de 1745 jusqu'à la fin de 1809, on y a baptisé 163499 garçons et 159405 filles, nombre dont le rapport est $\frac{39}{38}$, et diffère trop du rapport $\frac{25}{24}$ des baptêmes des garçons et des filles à Paris, pour être attribué au simple hasard.

30. Déterminons, d'après les principes précédens, les probabilités

A probabilistic program modeling the Laplace problem

```
def laplace(f1: int, g1: int, f2: int, g2: int) → float:
    p = sample(Uniform(0, 1), name="p")
    q = sample(Uniform(0, 1), name="q")
    observe(Binomial(f1 + g1, p), g1)
    observe(Binomial(f2 + g2, q), g2)
    return q > p

# Paris 1745 – 1784
fp = 377555
gp = 393386
# Londres 1664 – 1758
fl = 698958
gl = 737629
with ImportanceSampling(num_particles=100000):
    dist: Categorical = infer(laplace, fp, gp, fl, gl)
    print(dist.stats())
```

Conclusion

Expected knowledge and skills

Probabilistic Programming Languages (PPL)

Lecture notes (in french) ejcim.pdf

Take home

Inference is **intractable** in general.

Modelling a conditional random variable with a μPPL program and program hierarchical models.

Inference algorithms such as enumeration and importance sampling construct a discrete probability distribution that can represent exactly or approximate the law associated to the program.

Importance sampling follows Monte-Carlo simulation which produces correct approximation thanks to the Law of Large Numbers.

Next time:

Discrete Probabilistic Programs modeling, inference, syntax and semantics.